CPSC 2600 Assignment 3 Written Response

Jay Seung Yeon Lee
100357736

Using the dev tools, answer the following questions in a text file:

a. As you type the name of a new task, how is the component keeping track of what the user types in.

A: React application is keeping track of user typing by using useState hooks. Every time user enters something in the input box, react app is renewing the store state of the input field and keeping it updated.

b. When you click on the Edit button, you see a text box which allows you to update the name of an existing task. Which component is showing this template and how is it updating the name if you click on "Save".

A: Todo component is responsible for showing edit name template. Within Todo component, there are 2 templates prepared to be displayed: viewTemplate and editTemplate. When "Edit " button from view template or "Cancel" button from editTemplate is clicked, stored state will change which will allow users to display different templates accordingly. Notice how editTemplate is using a <form> element instead of <div> element used for viewTemplate. Form element allows users to add onSubmit event handler that will be triggered by button type "Submit", which in this case is the "Save" button. In our "handleSubmit" function, it will edit task using editTask function prop that has been passed up from App.js which will be rendered in .map() function to update the list of Todo components with newly edited name. And of course, revert the display of the component to viewTemplate by setting isEditing to false.

c. As you add new tasks to your do list, the list of tasks gets updated - Locate the component where this update is reflected

A: When user adds a new task to the list, component where this update is reflected should be in App.js component. Although user is adding a new task through Form.js component, all it's really doing is to read the user input written in the input field and pass it down to its parent component, App.js. App.js initially passes addTask function as a prop up to Form.js. And when the name of the task is passed back down to App.js, addTask function will update the state for tasks with all the pre-existing task names using the spread operator along with the new task name. Because we have a taskList method that maps to create Todo components, new task will be successfully created in the list of tasks in App.js.